Getekami Documentation

Release 0.3.0

Luis Felipe Lopez Acevedo, aka sirgazil

September 06, 2014

Contents

1 Contents **3**
- 1.1 Introduction . . . 3
- 1.2 Exercise Requirements . . . 4
- 1.3 Project Directory Layout . . . 5
- 1.4 Development Environment . . . 6
- 1.5 Development Process . . . 7
- 1.6 Deployment . . . 8

2 Indices and tables **11**

3 Public Domain Dedication **13**

Getekami is a Python GTK+ 3 Desktop Application mini-template created as an exercise on software development.

CHAPTER 1

Contents

1.1 Introduction

...

1.1.1 Target Platform

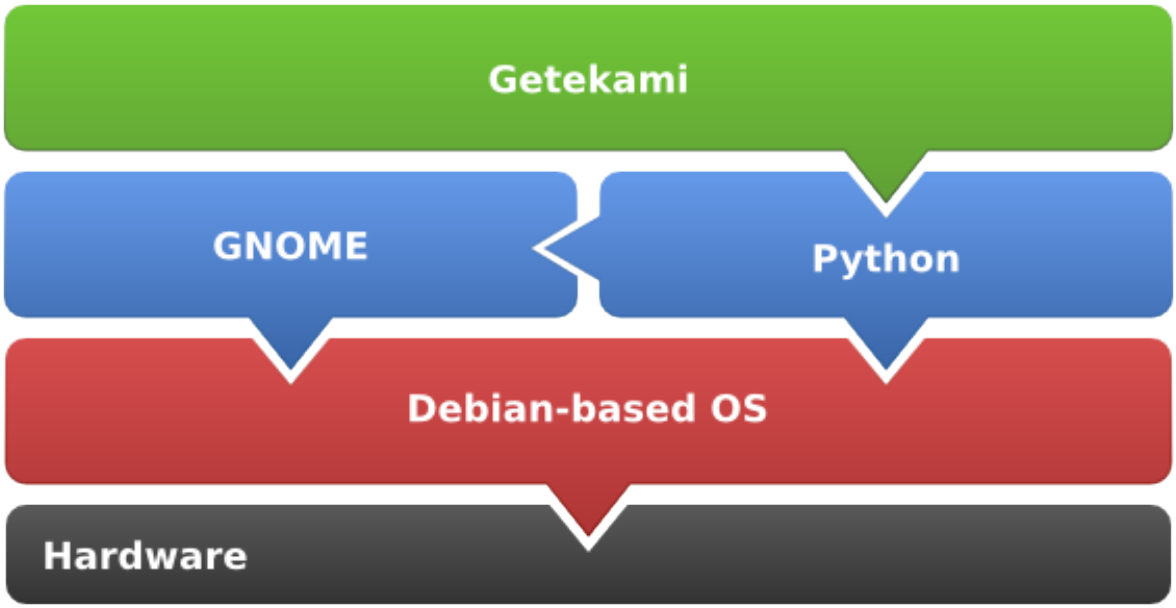

Figure 1.1: Getekami's target platform

Although, theoretically, Getekami can run on any platform where Python can run, the number of target platforms for this exercise has been reduced to the platform illustrated in the diagram above. The reason for this reduction is basically that the author does not support non-free platforms.

Getekami A desktop application mini-template written in Python.

Python A programming language, well integrated with the GNOME environment.

GNOME Desktop Environment A desktop environment that provides system integration, user interface, multimedia, communication, data storage and other utility libraries for desktop applications.

Debian-based Operating System Any libre operating system based on Debian.

Hardware Any computer where a Debian-based operating system can run.

1.1.2 See the final product

The resulting product of this exercise is a *desktop application* that does nothing in particular, but has basic integration with the GNOME desktop and serves as a template to start building your own applications.

Install *Getekami* to see for yourself. Run the following command as superuser:

```
# pip install getekami
```

And that's about it. Now you can run *Getekami* in any of the following ways:

- Go to **GNOME Activities** **Applications** **Development** and launch *Getekami*.
- Press **Alt+F2**, write *getekami*, and press **Enter**.
- In a terminal, run `$ getekami`.

1.2 Exercise Requirements

Project Management

- Use a distributed version control system (DVCS) to manage the project source files.
- Use a public hosting service for technology projects. The hosting service should provide, at least, the following items:
  - Integration with the distributed version control system (DVCS) of your choice.
  - Issue tracking. This is necessary to manage project tasks and bug reports.

Programming Language Use Python as the main programing language. The reasons for selecting this language are the following:

- You already use it.
- Easier to use than C, C++ or Vala.
- Widely used in introductory courses to programming.
- Available by default in most libre operating systems.
- Well integrated with the GNOME platform.

Graphical User Interface (GUI) Since the final product's target platform includes the GNOME Desktop Environment:

- Use GTK+ 3 for the GUI. GTK+ 3 components can be used in Python programs through PyGObject.
- Use Glade to create the GUI definition in XML. Using Glade has the following advantages:
  - Faster GUI development.
  - Better separation between application presentation and business logic.

  - The GUI generated with glade is independent from the programming language, which means that the same GUI can be used without change in applications written in other languages such as C, C++, Vala, etc.

- Use necessary files for integration with the GNOME Desktop Environment.
- Follow GNOME Human Interface Guidelines.
- Follow GNOME Accessibility Guidelines.

Software Architecture Use a well known software architectural pattern for the application.

User Manual

- Use Mallard for the application documentation (user manual). This is recommended, so that the documentation integrates with the GNOME Help Framework.
- Use GNOME Documentation Style Guide.

Development Process

- Use feature branches.
- Provide unit tests.
- Provide development documentation using Sphinx and RTD.

Packaging and Distribution

- Use an appropriate project directory structure for packaging and distributing the application as a Python package.
- Publish the final product in the Python Package Index (PYPI).
- Use of pip package manager as the recommended way of installing the final product.

1.3 Project Directory Layout

The following diagram illustrates the current directory layout and files of the project's source directory. The purpose of each directory and file is indicated in the list bellow.

```
getekami
|-- bin
|   `-- getekami
|-- data
|   |-- gui
|   |   `-- app-window.glade
|   `-- getekami.desktop
|-- docs
|   |-- source
|   |   |-- _static
|   |   |   |-- favicon.ico
|   |   |   |-- target-platform.png
|   |   |   `-- target-platform.svg
|   |   |-- _templates
|   |   |-- conf.py
|   |   |-- deployment.rst
|   |   |-- development_environment.rst
|   |   |-- development_process.rst
|   |   |-- exercise_requirements.rst
|   |   |-- index.rst
|   |   |-- intro.rst
```

```
|   |   `-- project_structure.rst
|   `-- Makefile
|-- getekami
|   |-- test
|   |   `-- __init__.py
|   |-- app.py
|   |-- app.pyc
|   |-- handlers.py
|   |-- __init__.py
|   `-- __init__.pyc
|-- help
|   |-- C
|   |   |-- figures
|   |   |   |-- logo.png
|   |   |   `-- logo.svg
|   |   |-- app-about.page
|   |   |-- doing-x.page
|   |   `-- index.page
|   `-- README.text
|-- CHANGES
|-- getekami.png
|-- getekami.svg
|-- MANIFEST
|-- packages-debian
|-- packages-python
|-- README
|-- README.text
`-- setup.py

12 directories, 36 files
```

getekami Project root directory.

getekami/bin Directory where all application binary files are located.

etc ...

1.4 Development Environment

Getekami is developed on the same platform described in the *Target Platform* page, with the addition of some other development tools. For this reason, the following instructions will indicate how to install only the additional development tools. It is assumed that you already have installed on your computer a libre Debian-based operating system with the GNOME Desktop Environment, and Python.

1.4.1 Development Packages

Glade A Rapid Application Development (RAD) tool that helps speed up the development of user interfaces for the GTK+ toolkit and the GNOME desktop environment.

Glade is used to design Getekami's graphical user interface and save it as an XML description.

pip A tool for installing and managing Python packages.

pip is the recommended package manager to install Getekami itself. It is also used to install some Getekami's dependencies.

PyGObject A Python dynamic module that enables developers to use GObject, which is an important part of the GNOME platform.

PyGObject is used by Getekami to use the GTK+ 3 user interface components.

Sphinx A tool that makes it easy to create intelligent and beautiful documentation.

Sphinx is used to write Getekami's development documentation.

1.4.2 Installation

Install required Debian packages by running the following command as a superuser while you are at Getekami's root directory.

```
# apt-get install $(< packages-debian)
```

Install required Python packages by running the following command as a superuser while you are at Getekami's root directory.

```
# pip install -r packages-python
```

1.5 Development Process

...

1.5.1 Starting Work on a New Version

- Create a new branch named after the version number (x.y.z).
- **Update version number in the source code and documentation.**
  - `docs/source/conf.py`.
  - `CHANGES`.
  - `README`.
  - `README.text`.
  - `setup.py`.
- **Update dependencies.**
  - `packages-debian`.
  - `packages-python`.
  - `setup.py`.
- Update classifiers in the `setup.py` file.

1.5.2 Finishing Work on a New Version

- Update CHANGES.
- Merge new version branch to the master branch.

1.6 Deployment

After the new version branch has been merged to the repository's default branch, it's time to package the application and distribute it to the public Web sites from where the final users can download and install automatically using the pip package manager.

1.6.1 Packaging the application

Run the following command from the project root directory:

```
$ python setup.py sdist
```

If everything goes right, a `dist` directory will be created and it will contain the packaged application ready for distribution. The package name should look like `getekami-x.y.z.tar.gz`, where x, y and z should match the version that will be released.

If you see the following warning while packaging the application, don't worry. The MANIFEST file will be created automatically:

```
warning: sdist: manifest template 'MANIFEST.in' does not exist (using default file list)
```

1.6.2 Testing the package

Before releasing the new version package to the world, install it locally using the package created above, try out the application, and uninstall. Make sure everything works as expected.

To install the application from the distribution package, run the following command in the root directory of the project:

```
# pip install ./dist/getekami-x.y.z.tar.gz
```

Once the application is installed, run it. You should see the application runs correctly:

```
$ getekami
```

Finally, uninstall:

```
# pip uninstall getekami
```

If no bugs were found during the testing, continue with the following section. Otherwise, report any bugs to the issue tracker, fix them, and start the deployment steps from the beginning.

1.6.3 Distributing the application

Now that the new version of the application is packaged and tested, it's time to upload it to public Web sites for distribution.

Upload to PYPI

This is the most important place to upload to since the Python Package Index (PYPI) is the standard repository for distributing third-party packages for Python.

If this is the first release of the application, you will have to register the package first in PYPI. To do so, run the following command from the project root directory:

```
$ python setup.py register
```

Once the package has been registered you will see that a project page has been created in PYPI for getekami: https://pypi.python.org/pypi/getekami.

Now, upload the new package version by running the following command in the project root directory:

```
$ python setup.py sdist upload
```

Visit the project page in PYPI to see the changes.

Upload to Bitbucket

Since the project source is hosted in Bitbucket, it's a good idea to add the new release to the *Downloads* section.

Go to the Downloads section and upload the application package located in the `dist` directory.

1.6.4 Release Announcement

At this point, you can go ahead and announce the package release in your Web site, Blog, user base mailing list, etc. You can use **Appendix A: Release Announcement Template**.

CHAPTER 2

Indices and tables

- *genindex*
- *modindex*
- *search*

CHAPTER 3

Public Domain Dedication

This work is dedicated to the public domain. You can copy, modify, distribute and use this work, for any purpose, all without asking permission. Do whatever you want with it.

Truly yours,

Luis Felipe Lopez Acevedo, aka sirgazil